CLAIMS

What is claimed is:

1. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can recover from a fault while it is communicating with a program on another of said computers across said communication network.

2. A method as recited in claim 1, wherein said transaction processing protects local data and processing against faults.

3. A method as recited in claim 1, wherein said replication protects processing and communication across said communication network against faults.

4. A method as recited in claim 1, wherein an object or process operates in a networked mode or a transactional mode.

5. A method as recited in claim 4:

wherein in said networked mode, an object or process on one computer can

interact with an object or process on an another computer across said communication network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

6. A method as recited in claim 4:

wherein in said transactional mode, an object or process on one computer can interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

7. A method as recited in claim 5:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another computer across said communication network.

8. A method as recited in claim 6:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

9. A method as recited in claim 8, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or aborts.

10. A method as recited in claim 1, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

11. A method as recited in claim 1, wherein messages sent between computers across said communication network are never retracted.

12. A method as recited in claim 1, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

13. A method as recited in claim 4:

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

14. A method as recited in claim 13:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

15. A method as recited in claim 14, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

16. A method as recited in claim 15, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

17. A method as recited in claim 15, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value from a database that is the same reply and value received during the initial operation of the object or process.

18. A method as recited in claim 4, wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

19. A method as recited in claim 18, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

20. A method as recited in 19, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

21. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can recover from a fault while it is communicating with a program on another of said computers across said communication network; and

wherein an object or process or process operates in a networked mode or a transactional mode.

22. A method as recited in claim 1, wherein said transaction processing protects local data and processing against faults.

23. A method as recited in claim 1, wherein said replication protects processing and communication across said communication network against faults.

24. A method as recited in claim 21:

wherein in said networked mode, an object or process or process on one computer can interact with an object or process on an another computer across said communication network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

25. A method as recited in claim 21:

wherein in said transactional mode, an object or process on one computer can interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

26. A method as recited in claim 24:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another computer across said communication network.

27. A method as recited in claim 25:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

28. A method as recited in claim 27, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or aborts.

29. A method as recited in claim 21, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

30. A method as recited in claim 21, wherein messages sent between computers across said communication network are never retracted.

31. A method as recited in claim 21, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

32. A method as recited in claim 21:

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

33. A method as recited in claim 32:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

34. A method as recited in claim 33, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

35. A method as recited in claim 34, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

36. A method as recited in claim 34, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value from a database that is the same reply and value received during the initial operation of the object or process.

37. A method as recited in claim 21, wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

38. A method as recited in claim 37, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

39. A method as recited in 38, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

40. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can recover from a fault while it is communicating with a program on another of said computers across said communication network;

wherein an object or process operates in a networked mode or a transactional mode;

wherein in said networked mode, an object or process on one computer can

interact with an object or process on an another computer across said communication network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

41. A method as recited in claim 40, wherein said transaction processing protects local data and processing against faults.

42. A method as recited in claim 40, wherein said replication protects processing and communication across said communication network against faults.

43. A method as recited in claim 40:

wherein in said transactional mode, an object or process on one computer can interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

44. A method as recited in claim 40:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another

computer across said communication network.

45. A method as recited in claim 43:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

46. A method as recited in claim 45, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or aborts.

47. A method as recited in claim 40, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

48. A method as recited in claim 40, wherein messages sent between computers across said communication network are never retracted.

49. A method as recited in claim 40, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

50. A method as recited in claim 40:

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

51. A method as recited in claim 50:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

52. A method as recited in claim 51, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

53. A method as recited in claim 52, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

54. A method as recited in claim 52, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value

from a database that is the same reply and value received during the initial operation of the object or process.

55. A method as recited in claim 40, wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

56. A method as recited in claim 55, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

57. A method as recited in 56, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

58. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can recover from a fault while it is communicating with a program on another of said computers across said communication network;

wherein an object or process operates in a networked mode or a transactional mode;

wherein in said transactional mode, an object or process on one computer can interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

59. A method as recited in claim 58, wherein said transaction processing protects local data and processing against faults.

60. A method as recited in claim 58, wherein said replication protects processing and communication across said communication network against faults.

61. A method as recited in claim 58:

wherein in said networked mode, an object or process on one computer can interact with an object or process on an another computer across said communication network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

62. A method as recited in claim 61:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another computer across said communication network.

63. A method as recited in claim 58:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

64. A method as recited in claim 63, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or aborts.

65. A method as recited in claim 58, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

66. A method as recited in claim 58, wherein messages sent between computers across said communication network are never retracted.

67. A method as recited in claim 58, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

68. A method as recited in claim 58:

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

69. A method as recited in claim 68:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

70. A method as recited in claim 69, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

71. A method as recited in claim 70, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

72. A method as recited in claim 70, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value from a database that is the same reply and value received during the initial operation of the object or process.

73. A method as recited in claim 58, wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

74. A method as recited in claim 73, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

75. A method as recited in 74, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

76. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can

recover from a fault while it is communicating with a program on another of said computers across said communication network;

wherein an object or process operates in a networked mode or a transactional mode;

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

77. A method as recited in claim 76, wherein said transaction processing protects local data and processing against faults.

78. A method as recited in claim 76, wherein said replication protects processing and communication across said communication network against faults.

79. A method as recited in claim 76:

wherein in said networked mode, an object or process on one computer can interact with an object or process on an another computer across said communication network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

80. A method as recited in claim 76:

wherein in said transactional mode, an object or process on one computer can

interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

81. A method as recited in claim 79:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another computer across said communication network.

82. A method as recited in claim 80:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

83. A method as recited in claim 82, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or

aborts.

84. A method as recited in claim 76, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

85. A method as recited in claim 76, wherein messages sent between computers across said communication network are never retracted.

86. A method as recited in claim 76, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

87. A method as recited in claim 76:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

88. A method as recited in claim 87, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

89. A method as recited in claim 88, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

90. A method as recited in claim 88, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value from a database that is the same reply and value received during the initial operation of the object or process.

91. A method as recited in claim 76, wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

92. A method as recited in claim 91, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

93. A method as recited in 92, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

94. A method for providing fault tolerance between computers of different enterprises across a communication network, comprising:

unifying transaction processing and object or process replication between computers across a communication network;

wherein a computer program operating on at least one of said computers can recover from a fault while it is communicating with a program on another of said computers across said communication network;

wherein an object or process operates in a networked mode or a transactional mode; and

wherein while an object or process is in transactional mode, a request received from another object or process that is not part of the same transaction is queued until the transaction commits or aborts.

95. A method as recited in claim 94, wherein said transaction processing protects local data and processing against faults.

96. A method as recited in claim 94, wherein said replication protects processing and communication across said communication network against faults.

97. A method as recited in claim 94:

wherein in said networked mode, an object or process on one computer can interact with an object or process on an another computer across said communication

network; and

wherein an object or process in networked mode is protected against faults by an object or process replication system.

98. A method as recited in claim 94:

wherein in said transactional mode, an object or process on one computer can interact with an object or process in a local database, but not with an object or process on another computer across said communication network; and

wherein data in transactional mode is protected against faults by a transaction processing system.

99. A method as recited in claim 97:

wherein an object or process operates in said networked mode by default; and

wherein in networked mode an object or process can interact freely with another object or process on the same computer, or with an object or process on another computer across said communication network.

100. A method as recited in claim 98:

wherein an object or process enters transactional mode either by explicitly initiating a transaction, or by being invoked by another object or process that is part of a transaction; and

wherein in transactional mode, an object or process cannot invoke methods of a

remote object or process across said communication network, and cannot accept request messages that originate from an object or process that is not part of the transaction.

101. A method as recited in claim 100, wherein an object or process leaves transactional mode and returns to networked mode when the transaction commits or aborts.

102. A method as recited in claim 94, wherein computers of different enterprises across said communication network maintain consistent views of interactions across said communication network.

103. A method as recited in claim 94, wherein messages sent between computers across said communication network are never retracted.

104. A method as recited in claim 94, wherein a fault in a computer of one enterprise will not abort activities in a computer of another enterprise.

105. A method as recited in claim 94:

wherein roll-forward recovery is used in networked mode; and

wherein roll-back/abort recovery is used in transactional mode.

106. A method as recited in claim 105:

wherein said roll-forward recovery starts from a checkpoint and then replays messages from a message log; and

wherein messages involved in an aborted transaction are not replayed.

107. A method as recited in claim 106, wherein roll-forward recovery of one object or process does not disrupt continued operation of another object or process or of a database.

108. A method as recited in claim 107, wherein a message generated during roll-forward recovery of an object or process is detected as a duplicate message and is not processed a second time.

109. A method as recited in claim 106, wherein an object or process recovered using roll-forward recovery receives a reply from another object or process and a value from a database that is the same reply and value received during the initial operation of the object or process.

110. A method as recited in claim 94, wherein recovery of an object or process restores the state of the object or process and then processes a message that was queued waiting for a transaction to commit or abort.

111. A method as recited in 110, wherein a message for a current transaction is processed but a message of an enclosing transaction or no transaction remains queued.

5